

Mechanical generation of networks with surplus complexity

Russell K. Standish

Mathematics and Statistics, University of New South Wales
hpcoder@hpcoders.com.au

Abstract. In previous work I examined an information based complexity measure of networks with weighted links. The measure was compared with that obtained from by randomly shuffling the original network, forming an Erdős-Rényi random network preserving the original link weight distribution. It was found that real world networks almost invariably had higher complexity than their shuffled counterparts, whereas networks mechanically generated via preferential attachment did not. The same experiment was performed on foodwebs generated by an artificial life system, Tierra, and a couple of evolutionary ecology systems, EcoLab and WebWorld. These latter systems often exhibited the same complexity excess shown by real world networks, suggesting that the *complexity surplus* indicates the presence of evolutionary dynamics. In this paper, I report on a mechanical network generation system that does produce this complexity surplus. The heart of the idea is to construct the network of state transitions of a chaotic dynamical system, such as the Lorenz equation. This indicates that complexity surplus is a more fundamental trait than that of being an evolutionary system.

1 Introduction

This work situates itself firmly within the *complexity is information content* paradigm, a topic that dates back to the 1960s with the work of Kolmogorov, Chaitin and Solomonoff. Indeed, the seminal work of Mowshowitz in 1968 [5,6,7,8] describes an information-based network complexity measure called *graph entropy*, that is essentially a generalisation of the measure presented here, work that has by and large been forgotten by the complex systems community, only to be reinvented in recent times[1].

The idea is fairly simple. In most cases, there is an obvious *prefix-free* representation language within which descriptions of the objects of interest can be encoded. There is also a classifier of descriptions that can determine if two descriptions correspond to the same object. This classifier is commonly called the *observer*, denoted $O(x)$.

To compute the complexity of some object x , count the number of equivalent descriptions $\omega(\ell, x)$ of length ℓ that map to the same object x under the agreed classifier. Then the complexity of x is given in the limit as $\ell \rightarrow \infty$:

$$\mathcal{C}(x) = -\log P(x) = \lim_{\ell \rightarrow \infty} (\ell \log N - \log \omega(\ell, x)) \quad (1)$$

where N is the size of the alphabet used for the representation language. Loosely speaking, $P(x)$ here is the probability that a description chosen uniformly at random will describe the object x .

To fix the representation language of graph using a binary alphabet ($N=2$), we start with a prefix of $\lceil \log_2 n \rceil$ 1s followed by a ‘0’ stop bit. This indicates the number of bits needed to store n , the number of nodes. Next we encode the number of links l , which for a directed graph requires $\lceil \log_2 n + \log_2(n-1) \rceil$ bits. Finally, we encode the linklist, as a *rank index* within the

$$\Omega = \binom{L}{l}$$

possible linklists with l links, where $L = n(n-1)$. So each description of an n -node, l -link graph is precisely $\ell_{n,l} = 1 + 2\lceil \log_2 n \rceil + \lceil \log_2 n + \log_2(n-1) \rceil + \lceil \log_2 \Omega \rceil$ bits long. Descriptions longer than that represent the same graph as the initial leadin sequence just described, with the trailing bits irrelevant, thus $\omega(\ell, x) = \omega(\ell_{n,l}, x)2^{\ell - \ell_{n,l}}$. Substituting into equation (1) gives (in bits):

$$\mathcal{C}(x) = \ell_{n,l} - \log_2 \omega(\ell_{n,l}, x). \quad (2)$$

The relationship of this algorithmic complexity measure to more familiar measures such as Kolmogorov (KCS) complexity, is given by the coding theorem[3, Thm 4.3.3]. In this case, the descriptions are halting programs of some given universal Turing machine U , which is also the classifier. Equation (1) then corresponds to the logarithm of the *universal a priori probability*, which is a kind of formalised Occam’s razor that gives higher weight to simpler (in the KCS sense) computable theories for generating priors in Bayesian reasoning. The difference between this version of \mathcal{C} and KCS complexity is bounded by a constant independent of the complexity of x , so these measures become equivalent in the limit as message size goes to infinity.

In setting the classifier function, we assume that only the graph’s topology counts — positions, and labels of nodes and links are not considered important. Links may be directed or undirected and can have a positive real number weight attached to them.

If all links have the same weight, then the counting problem of determining $\omega(\ell_{n,l}, x)$ for networks turns out to be equivalent to the automorphism group problem of determining if two graphs are automorphic. Whilst this problem is suspected of being combinatorially hard[4], several practical algorithms are available for computing the size of the automorphism group for reasonable sized networks of thousands of nodes.

To handle weighted links, the idea is to interpolate between the graph with the link, and without the link, according to the link’s weight. The algorithm performs a weighted sum of graph complexity over the different weights w present in the network, of graphs composed of links of weight greater than or equal to w . Details can be found in [11].

In [11], this measure is applied to a number of well-known naturally occurring networks, mostly published food webs. The measure is also compared with

an ensemble of networks obtained by shuffling the link structure, with a positive complexity difference (*complexity surplus*) usually found, indicating it is measuring something structurally important about the network. The same technique was applied to Edös-Rényi generated networks, which unsurprisingly had no complexity surplus, and to networks generated via preferential attachment, which also generates no significant complexity surplus.

In [10], I made the observation that artificial evolutionary processes, such as EcoLab and Tierra also did not lead to a complexity surplus, making this property of natural networks a mysterious one. However, this later turned out to be due to a bug in the analysis[11], and when corrected, led to significant complexity surpluses being generated for EcoLab and Tierra, though not for WebWorld, another evolutionary ecology model similar to EcoLab.

In this paper I report on another (non-evolutionary) mechanical technique for generating networks that does generate significant amounts of complexity surplus.

2 Generating networks from timeseries data

Michael Small has been analysing timeseries data by generating networks that represent the dynamics behind the timeseries, and then applying network analysis techniques to tease out features of the data. His most recent version of the technique[9], which I shall describe below, can be applied to any timeseries, continuous or discrete, and of any dimension. Of particular interest are timeseries derived from chaotic dynamics, which generate particularly beautiful filigree networks, that are prime candidates for high structural complexity.

Consider first a discrete timeseries (x_1, x_2, \dots, x_n) where $x_i \in X \subset \mathbb{Z}$. Each element of X labels a node of the target network, and links of the network are weighted by the number of transitions (x_i, x_{i+1}) in the timeseries between the labels of source and target nodes of the network link. Concretely, consider a simple timeseries 1,2,3,1,2,1. The generated network is shown in figure 1, where the link $1 \rightarrow 2$ has double the weight of the other links.

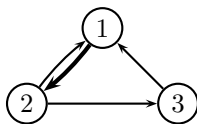


Fig. 1. Example network generated from the timeseries 1,2,3,1,2,1.

If the timeseries data is continuous rather than discrete, then a means to convert real-valued data into integer labels is required. In this paper, I use a simple coarse graining, where the number of cells in each dimension of the timeseries is

chosen to give a target network size. For example, with three dimensional data, one can choose 10 cells along each dimension to give a target network size of 1000 nodes.

It should be pointed out that Small gives a more sophisticated approach aimed at extracting the maximum dynamic information from the timeseries, which involves choosing a window size w , and ranking the values within the window to give a well defined sequence of integers. For example, the sequence (0.5, 0.2, 0.35, 0.4, 0.3) will for $w = 3$ give the sequence of ranks: (3,1,2), (1,2,3), (2,3,1). The generated networks will have $m(w) \leq w!$ nodes. Finally, the parameter w is not arbitrary, but chosen to maximise the amount of information captured. The curve $m(w)$ is roughly sigmoidal in shape, and the optimum value w_{opt} occurs at the point of inflection, where $\Delta m(w_{\text{opt}})$ is maximised.

3 Results

Dataset	nodes	links	\mathcal{C}	$e^{\langle \ln \mathcal{C}_{\text{ER}} \rangle}$	$\mathcal{C} - e^{\langle \ln \mathcal{C}_{\text{ER}} \rangle}$	$\frac{ \ln \mathcal{C} - \langle \ln \mathcal{C}_{\text{ER}} \rangle }{\sigma_{\text{ER}}}$
celegansneural	297	2345	442.7	251.6	191.1	29
PA1	100	99	98.9	85.4	13.5	2.5
Lorenz	8000	62	560.2	56.0	504.2	58.3
Hénon-Heiles	10000	31	342.0	57.3	284.7	55.6

Table 1. Node and link count, network complexity, average shuffled complexity, surplus and the number of standard deviations of the shuffled distribution that the surplus represents. Listed here are the well-known *C. elegans* neural network, preferential attachment with outdegree 1 (from [11]), and networks generated from the Lorenz and Hénon-Heiles systems.

The first study looked at applying Small’s network generation technique to the dynamics of a couple of well-known chaotic dynamical systems — the Lorenz system, given by equation (3) and the Hénon-Heiles system, given by equation (4).

$$\begin{aligned}
\dot{x} &= \sigma(y - x) \\
\dot{y} &= x(\rho - z) - y \\
\dot{z} &= xy - \beta z
\end{aligned}
\tag{3}$$

$\sigma = 10, \rho = 28, \beta = 8/3$

$$\begin{aligned}
\ddot{x} &= -x - 2xy \\
\ddot{y} &= -y - (x^2 - y^2)
\end{aligned}
\tag{4}$$

The networks were generated according to the scheme described in §2. Shown in table 1 are the node and link counts of the generated network, the complexity

value \mathcal{C} computed according to eq (1), the average complexity value of shuffled networks, the difference between the complexity and the average shuffled complexity (ie surplus) and finally the number of standard deviations of the shuffled distribution that the surplus corresponds to. It is reported this way, as the p value is far too ridiculously small to be comprehensible. For more detailed discussion of the analysis, please refer to [11]. Also shown, for comparison, are a couple of results from that paper.

The next experiment performs the same analysis on timeseries generated by 1D binary cellular automata with a neighbourhood of 3. The number of cells chosen was 10, so the generated network will have 1024 nodes (2^{10} possible states). The CAs were initialised to a random state, and the first 1000 steps discarded to eliminate transient effects. Figure 2 shows the complexity, and average shuffled complexity plotted as a function of Langton's λ [2], for all 256 neighbourhood 3 binary CAs. What is clear is that there a large spike in the complexity values (and in the surplus) around $\lambda = 0.5$, corresponding to chaotic rules.

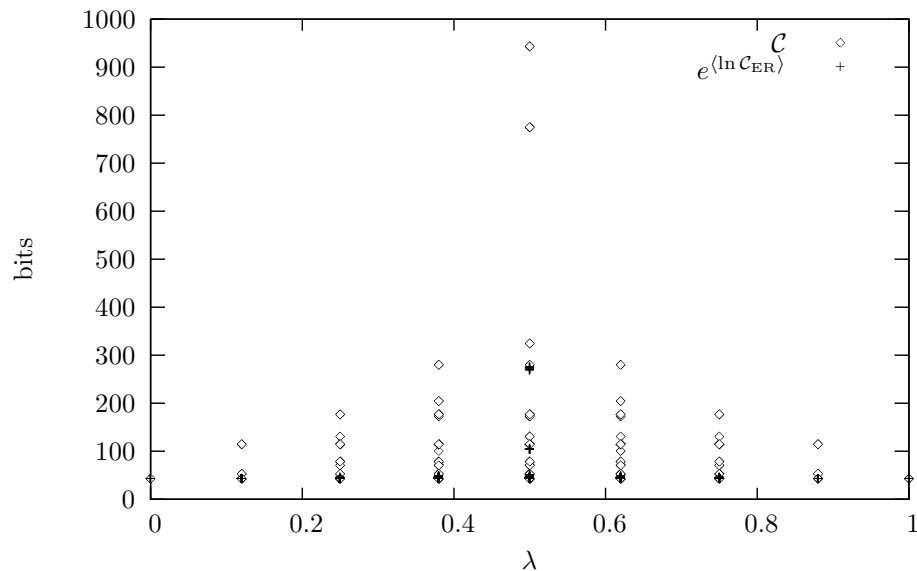


Fig. 2. Plot of \mathcal{C} and averaged shuffled \mathcal{C} for networks generated from all 256 1D CAs with a 3 cell neighbourhood, plotted as a function of Langton's λ .

Source code used for these experiments can be found as part of *Ecophab* version 5.D19, available from <http://ecolab.sourceforge.net>.

4 Discussion

Generating networks from chaotic dynamical systems proves to be a good means of generating networks with complexity surplus. We can now begin to study systems that can be tuned across a range of behaviour from ordered to chaotic. A preliminary experiment showed that chaotic behaviour is associated with the presence of large amounts of network complexity. One might suspect that the complexity surplus is associated with complex dynamics, or with Wolfram class 4 cellular automata. The preliminary experiment reported here did not specifically support that, as the peak occurred for chaotic (class 3) cellular automata, however the class 4 regime tends to be a very small part of the λ spectrum, so I doubt that enumerating all 1D 3-neighbourhood CA can resolve the issue. Instead, transition to chaos experiments, such as Langton's original study with a 4-state, 5-neighbourhood CA[2] will probably be more suitable.

Acknowledgments

I wish to thank Michael Small for helpful discussions on this topic.

References

1. Dehmer, M., Mowshowitz, A.: A case study of cracks in the scientific enterprise: Reinvention of information-theoretic measures for graphs. *Complexity* (2014), DOI: 10.1002/cplx.21540
2. Langton, C.G.: Computation at the edge of chaos: Phase transitions and emergent computation. *Physica D* 42, 12–37 (1990)
3. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and its Applications. Springer, New York, 2nd edn. (1997)
4. Lubiw, A.: Some NP-complete problems similar to graph isomorphism. *SIAM Journal on Computing* 10(1), 11–21 (1981), <http://link.aip.org/link/?SMJ/10/11/1>
5. Mowshowitz, A.: Entropy and the complexity of graphs: I. an index of the relative complexity of a graph. *Bulletin of Mathematical Biology* 30(1), 175–204 (1968)
6. Mowshowitz, A.: Entropy and the complexity of graphs: II. the information content of digraphs and infinite graphs. *Bulletin of Mathematical Biology* 30(2), 225–240 (1968)
7. Mowshowitz, A.: Entropy and the complexity of graphs: III. graphs with prescribed information content. *Bulletin of Mathematical Biology* 30(3), 387–414 (1968)
8. Mowshowitz, A.: Entropy and the complexity of graphs: IV. entropy measures and graphical structure. *Bulletin of Mathematical Biology* 30(4), 533–546 (1968)
9. Small, M.: Complex networks from time series: Capturing dynamics. In: *IEEE International Symposium on Circuits and Systems Proceedings*. pp. 2509–2512 (2013)
10. Standish, R.K.: Network complexity of foodwebs. In: Fellerman, H., et al. (eds.) *Proceedings of Artificial Life XII*. pp. 337–343. MIT Press, Cambridge, MA (2010)
11. Standish, R.K.: Complexity of networks (reprise). *Complexity* 17, 50–61 (2012), arXiv: 0911.3482